REMARKS

The Examiner rejected claims 1, 5, 9, 14, and 18 under 35 U.S.C. §112, second paragraph as indefinite by use of the "substantially". This problem is removed since all the claims now recite that the new and stored software components are combined without changing any code within the software components and without writing any adapters. This language is consistent with the specification language at page 8, line 32 stating:

> "As set forth above, the present invention provides an approach for combining independent, building blocks (components) into larger applications without changing any code within the building blocks and without writing any adapters."

The Examiner rejected claims 1-18 under 35 U.S.C. §103 as unpatentable over Foody.

At page 3 of the Office Action, the Examiner stated:

> "...the limitations of the claims... is silent in regards to the software components of the new software that do not have inputs and outputs of the same name. The Examiner submits that the code is changed when the software components of the new software do not have inputs and outputs of the same name".

Applicants submit that this is not a problem, because claim 1 internally recites that the new software component inputs and outputs *are "all"* automatically linked to the inputs and outputs of the same name of said stored software components. In this case, as the specification makes abundantly clear, with this condition there is no changing of code within the software components and the combination of the new and stored software components occurs without writing any adapters.

Claim 1 distinguishes over Foody in a number of ways. First, claim 1 recites that the software components are dynamically loadable at runtime and which have dynamically linkable named inputs and outputs stored on a memory of the computer system. For support of this language, see Applicants' specification at page 6, lines 16-20, for example.

Claim 1 next distinguishes over Foody at least by reciting said components also having internal tasks for queuing of data transferred into and out from the components via said inputs and outputs. This internal task language was in original claim 5, for example.

Claim 1 next distinguishes over Foody at least by reciting an event communication framework providing automated pattern-based fully distributable events such that when a new dynamically loadable at runtime software component is loaded into said computer system also having dynamically linkable named inputs and outputs, the new software component inputs and outputs are all automatically linked to the inputs and outputs of the same name of said stored software components so that the new and stored software components are combined without changing any code within the software components and without writing any adapters. This language is supported, for example, by specification disclosure at page 5, lines 24-28.

Foody only discloses a software adapter for various communication infrastructures such as, for example, CORBA, COM, or DSOM. Foody does not specify any event communication framework providing automated, pattern-based fully distributable events, and does not disclose combining of the new software component without changing any code within the software components and without writing any adapters.

In Foody, the event determines operations.

Foody does not provide a dynamic linking of all named inputs and outputs of the new component and also does not have internal tasks for queuing of data transferred into and out from the components via said inputs and outputs.

In Foody, the objects specified have a hard link dependency. Thus there is no automatic linking of all of the inputs and outputs of a new software component to the inputs and outputs of the same name of a stored software component.

As Foody states at claim 3 (column 231, lines 38-39), he is only "adding or removing support for object systems and object models"). Thus Foody is only a protocol-adapter and not a software component.

At column 12, lines 32-36, Foody indicates that descriptions of class and type will be combined to describe a complete class. These elements will be signed if they are semantic significant or not. Both kinds of information will be integrated in one description. This disclosure has nothing to do with claim 1 where the software component is able to connect all of the same name inputs and outputs of the new component at runtime regardless of exchanged data.

Foody at column 1, lines 60-67 describes C++ calling mechanisms – direct or indirect calls. This does not relate to automatic linking of all same name inputs and outputs of the new component

Foody at column 2, lines 1-5 describes a handoff of C function pointers and C++ member function pointers, which is a mechanism of the programming languages C and C++ respectively. This does not relate to automatic linking of all same name inputs and outputs of the new component.

Foody at column 10, lines 10-18 says that the enumeration framework makes a sweep of generic capabilities available over-writable by object systems with their own capabilities. See also Foody column 10, lines 39-49. This does not relate to dynamic linking of all same named inputs and outputs automatically of the new component.

Foody at column 9, lines 1-27 locates objects in an object specific name space. This describes the localization of objects in "name space" of their object systems. This has nothing to do with event communication framework with automatic dynamic linking of all same named inputs and outputs of the new component.

Dependent claims 2-4 distinguish at least for the reasons claim 1 distinguishes and also by reciting additional features not suggested.

Independent claim 5 distinguishes at least for the reasons claim 1 distinguishes at least for the reasons claim 5 distinguishes and also by reciting additional distinguishable features.

Independent claim 9 distinguishes at least for the reasons claim 1 distinguishes.

Claim 10 distinguishes at least for the reasons claim 1 distinguishes. Dependent claims 11-13 distinguish at least for the reasons that claim 10 distinguishes and also by reciting additional distinguishing features.

Independent claim 14 distinguishes at least for the reasons claim 1 distinguishes. Dependent claims 15-17 distinguish at least for the reasons claim 14 distinguishes and also by reciting additional features not suggested.

Independent claim 18 distinguishes at least for the reasons noted with respect to claim 1.

Allowance of the case is respectfully requested.

Respectfully submitted,

(Reg). #27,841

Brett A. Valiquet
Schiff Hardin LLP
Patent Department
6600 Sears Tower
Chicago, Illinois 60606
Telephone: 312-258-5786
Attorneys for Applicants
CUSTOMER NO. 26574

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as First Class Mail in an envelope addressed to: Commissioner for Patents, P. O. Box 1450, Alexandria, VA 22313 on January 5, 2006.

BRETT A. VALIQUET